APPENDIX A

In one typical MDM (Multiple Device Management) implementation, the schema interface may be derived from the Common Information Model (CIM) standard for representing computing process and devices. Such an implementation on a Microsoft® Windows® operating system, for example, would be referred to as a WMI schema. The various information herein describes such a suitable schema and other related data, but should be considered as only an example.

Identifier Definitions

SET NAME

The set name uniquely identifies a set. It is stored in Sets.Name property and can be up to 256 Unicode characters. Any character is valid, but non-printable characters are not recommended. Names are case-insensitive, but case is preserved. The set name needs to be unique across all set names.

DEVICE NAME

The device name uniquely identifies a device, and is used to find the IP address of the device in order to communicate with it. It is stored in Devices.Name property. It can be up to 256 Unicode characters. Any character is valid, but non-printable characters are not recommended.

Names are case-insensitive, but case is preserved. The set name needs to be unique across all device names. Note that device name and set name have the same number of characters because they are stored in a single field in the JobInvocations object.

JOB NAME

The job name is used to uniquely identify job templates, and as a piece of identification information for job history records. It is stored in the JobInvocations.Name and Jobs.Name properties. The job name can be up to 50 Unicode characters. With any character being valid, but non-printable characters not recommended. Names are case-insensitive, but case is preserved. For job templates (jobs where the job identifier is zero), it needs to be unique. However the same name may also be used, possibly multiple times, in job history records.

DEVICE TYPE NAME

The device type name is used to identify the type of a device. It is stored in the Devices.Type property, and defined in the values of DeviceTypes.Name properties. It can be up to 50 Unicode characters. All characters are valid, but non-printable characters are not recommended. Names are case-insensitive, but case is preserved. It needs to be unique across all instances of DeviceTypes.

DESCRIPTIONS

Descriptions are typically 256 Unicode characters (but see individual object definitions for specific values). Any Unicode characters are valid, including carriage returns and newlines.

Objects

Each object may contain properties and methods, and have associations with other objects. In one implementation, the objects, properties, methods and associations are exposed through WMI. Additional association classes are used to implement associations, as described below.

Each object definition starts with a summary, then a definition in the following format:

Class name	Object name: this is the class name in WMI class name.
Derived from	WMI class name that this class is derived from
Description	Description text of this class, as stored in WMI

The conditions under which the class can be created and deleted are described below, along with information about the other classes to which this class is associated. Each property of the class is also listed, in this format:

Property name	Name of the property in WMI
Description	Description text of the property, as stored in WMI
Type	CIM type of the property
Access	Whether the property can be read, written to or both
Key	Whether this property is a key value. Multiple properties can be a key.
Values	Description of value of the property, such as maximum length of a string property or meaning of the numbers of an integer property.

Each method is described as set forth below:

Description	Description of the method, as stored in WMI	
Arguments:		
Argument name	Direction of the argument (in or out), and CIM type	Description text of the argument, as stored in WMI
Return value	CIM type	Meaning of the return value

Sets

In essence, sets are the building blocks of multiple device management, and comprise objects that group devices according to some logical or physical grouping. There is a one-to-many relation between a Sets object and Devices objects. Commands can be executed on each member of a Sets object by means of the Execute method.

Class name	Sets
Derived from	CIM LogicalElement
Description	"The Sets class represents a collection of devices. Sets group devices for administrative purposes, and run jobs on multiple devices simultaneously. Each set must have a unique name. Sets can contain devices but not (at least presently) other sets."

Instances of the Sets class are created by the user (or other process). In one implementation, instances are never created automatically by the object model. The only property required for new instances of the Sets class is Name. The value of Name needs to be unique among the instances of Sets on the controller.

Instances of the Sets class can be deleted by the user (or other process). Instances are never deleted automatically by the object model in a described implementation. When an instance is deleted, no changes are made to other instances (including those instances which refer to the class being deleted).

Instances of Sets can have associations to instances of Devices; an instance of Sets has an association to each instance of Devices which is a member of the set. Each instance of Sets many be associated with zero, one or more instances of Devices. Instances of Sets can also have associations to instances of JobInvocations. If the JobInvocations instance is a template (the RootJobID property of JobInvocations is zero), then the associate represents that the job template represented by the JobInvocations instance is to be run on the associated set. If the JobInvocations instance is a history record (RootJobID is not zero), then the associate represents the fact that the job in JobInvocations was run on the associated set.

Set Properties

Name

Property name	Name
Description	"Name of the set"
Type	String
Access	Read
Key	Yes
Values	256 Unicode characters, excluding the newline or carriage return characters

The value of this property can be changed using the Rename method. The value must be unique among the Sets instances on the system.

Description

Property name	Description
Description	"Description of the set"
Type	String or NULL
Access	Read, Write
Key	No
Values	256 Unicode characters

This is a free-text description of the set. It may be NULL or empty, and can be updated.

Unused Properties

The Caption, InstallDate and Status properties are inherited from the parent classes but are currently unused in a current implementation.

Methods

AddDevice

AddDevice is used to add controlled or uncontrolled device to a set.

Description	"The AddDevice method adds a device as a member of the set."	
Arguments:		
Device	[in] Devices	"The Device input parameter is the reference to device. It represents the path of the devices which is to be associated with a set."
Return value	Void	

RemoveDevice

Description	"The RemoveDevice method removes a device from the set."
Arguments:	

Device	[in] Devices	"The Device input parameter is the reference to devices. It is the path of the device which is to be disassociated from the set."
Return value:	Void	

Rename

Description:	"The Rename method renames the set name to the name specified."	
Arguments:		
SetName	[in] string	"The SetName input parameter provides the name for the set.".
Return value:	Void	

Execute

Description:	"The Execute method runs a job on the devices in a set. If the job started successfully, the method returns the job identifier of the parent job.	
Arguments:		
JobInvocationName	[in] string	"JobInvocationName input parameter is the name that will be assigned to the Name property of the JobInvocations instance that will be created."
CommandType	[in] 32 bit integer	"CommandType determines how the command parameter is interpreted" [DCR 4880 added this argument]
Command	[in] string	"The Command input parameter is the path of the command to be run. The values that may be given depend on the value of the CommandType parameter."
Parameters	[in] string	"The Parameters input parameter specifies the arguments given when the job is started. This parameter is used depending on the value of the CommandType parameter."
Description	[in] string	"Description input parameter is the description for the executing command which is to be logged in the JobInvocations object."
JobID	[out] 64 bit integer	If no error occurs (the return value is 0), this contains the RootJobID of the newly created JobInvocation instance. If an error occurs (the return value is not zero), this contains 0. [DCR 4891 added this argument]
Return value:	32 bit integer	The RootJobID of the newly created JobInvocation instance, or 0 if the job was not started.

This method causes a job to be run against the members of the set on which it is run. If the job is successfully created, it causes a JobInvocations instances to be created for the job. The value of the RootJobID property of the new JobInvocations instance is returned as the return value of this method. Note that job execution is asynchronous, so success of this method does not mean that the job itself will be successful on the agents.

The CommandType argument specifies how the Command and Parameters fields are to be interpreted:

CommandType value	**Meaning of Command argument**	**Meaning of Parameters argument**
1	Specifies a script to be downloaded from the controller to the devices. Must contain a file that is accessible from the controller (either a local path on the controller or a UNC path).	Command line parameters for the job
2	Specifies a script or binary accessible from both the controller and the agents. Agent accesses the file using the path given (either local path on the agent or a UNC path)	Command line parameters for the job
3	Special command. See available commands, listed below	Ignored

This is written to the Command property of the new JobInvocations instances. The value of the Parameters argument field is silently ignored if CommandType is 3. The maximum length of the Description argument is 256 Unicode characters. This is stored in the Description property of the new JobInvocations instance. It is not an error for the set to be empty. In this case, a JobInvocations instance is created as normal, along with the parent Jobs instance. However there are no child Jobs instances. Users of the object model must be prepared for this situation.

Special Commands

If CommandType is 3, then the Command argument contains one of the following values:

Contents of the Command argument	**Meaning**
"Shut down"	Shutdown the server
"Reboot"	Reboot the server

The text is case-insenstive (that is, for example, "Reboot", "REBOOT" and "reboot" all cause the server to reboot). If the value of the Command argument is not one of these, the method returns an error.

DEVICES

Devices are members of sets; sets are groups of devices. Devices represent physical computer systems which for example are typically server appliances. With multiple device management, one goal is to perform management on many machines simultaneously. Nevertheless, with the device object commands can also be executed.

Class name	Devices
Derived from	CIM UnitaryComputerSystem
Description	"The Devices class represents the devices that are either automatically discovered or manually added."

Creation

Instances of Devices can be created at a request of the user or other process, or automatically by the controller based on the reception of an auto-discovery packet from an agent. An instance of the Devices class may be created manually by the user or other process. The new instance needs at least a value for the Name property, and this value cannot already exist as the Name of another instance of the

Devices class on the system. The value of the Name property needs to be a name that can be resolved on the controller to the IP address of the administrative interface on the device itself. Typically this resolution will occur using a DNS server.

The new instances can be associated with one or more MAC addresses, and each MAC address with one or more IP addresses. This is indicated by associating the new instance of the Devices class with instances of DeviceHWAddrs, and associations of DeviceHWAddrs with DeviceHWIPAddrs. However, only the name is needed to communicate with machine. The controller uses DNS to resolve the name to an IP address. MAC and IP are not used for communication and are essentially for information only on the controller. When the instance is created, controller sends request to agent to get node information, including the IP and MAC addresses. For all manually created Devices instances, the LastDiscoveryTime will be NULL. If at some later point an auto-discovery packet is received that matches this Devices instance, this field will be updated with the time that the packet was received.

The SMBIOS GUID alternatively may be stored in the controller and used to uniquely identify the device. This can be done with no modifications to the object model, since it can be stored as a new device type (in DeviceHWAddrs, linked to a DeviceTypes representing the SMBIOS GUID.

Deleting

Instances of the Devices class may be deleted by the user or other process. Instances may be deleted by the system if the Controller.RefreshDeviceList() method is called. When an instance of Devices is deleted, any instances of DeviceHWAddrs that refer to the same device as also deleted (which might also cause deletion of linked DeviceHWIPAddrs

Associations

Instances of Devices can have associations to the following classes:

- To instances of Sets

  An instance of Devices has an association to each instance of Sets of which it is a member. Each instance of Devices many be associated with zero, one or more instances of Sets.

- To instances of JobInvocations

  An instance of Devices has an association to JobInvocations. If the JobInvocations instance is a template (the RootJobID property of JobInvocations is zero), then the associate represents that the job template represented by the JobInvocations instance is to be run on the associated device. If the JobInvocations instance is a history record (RootJobID is not zero), then the associate represents the fact that the job in JobInvocations was run on the associated device.

- To instances of Jobs

  An instance of Device has an association to instances of Jobs that represents the parent job of each job run on this device.

- To instances of DeviceHWAddrs

  An instance of Device has an association to an instance of DeviceHWAddrs for each hardware address on the device that the controller knows about. In a current version, the only hardware addresses stored on the controller are MAC addresses of NIC cards.

- To a single instance of DeviceTypes

  An instance of Devices has an association to an instance of DeviceTypes giving the device type. In a current version, only a single device type is supported, so all Devices instances associate with a single instance of DeviceTypes.

Properties

Name

This is inherited from the parent class.

Property name	Name
Description	<inherited>
Type	String
Access	Read
Key	Yes
Values	256 Unicode characters

The value of this property can be changed using the Rename method.

Alive

Property name	Alive
Description	"Status flag to indicate whether the controlled device is alive."
Type	Boolean
Access	Read
Key	No
Values	True if device is alive; false if device is not alive

If the device is uncontrolled, this property is always false. Otherwise, this property is set to true when a device becomes controlled, and set to true (if currently set to false) every time a heartbeat is received from the device. The heartbeat time period is a global setting for the controller. The agent should send a heartbeat packet every heartbeat time period. The controller will set this property to false if it does not receive a heartbeat from the device within a period of one-and-a-half times the heartbeat time period. The heartbeat time period used is the one currently set on the controller. This may be different to the device's heartbeat time period.

Controlled

Property name	Controller
Description	"Status flag to indicate whether the device is being controlled."
Type	Boolean
Access	Read
Key	No
Values	True if device is controlled from this controller; false if not

The value of this property is set by starting or ending control of the device, using the Manage method.

HeartBeatTime

Property name	Heartbeat
Description	"Time at which heartbeat from the controlled device is updated. This property is periodically updated by the heartbeat communication between the controller device and controlled device."
Type	Datetime or NULL
Access	Read
Key	No
Values	

If the device is uncontrolled or becomes not controlled, this is NULL. Otherwise, when a device becomes controlled this is set to the time that it becomes controlled. While a device is controlled, this is set to the time that the most recent heartbeat was received from the device.

AlertStatus

Property name	AlertStatus
Description	"Indicates whether alert- triggering mechanism is enabled, disabled, unavailable., status not known"
Type	32 bit integer
Access	Read
Key	No
Values	0=Disabled, 1=Enabled, 2=Unavailable, 3=Status Not Known

If the device is not controlled or becomes not controlled, this is always set to Unavailable. Otherwise, when a device becomes controlled this is initially set to Status not Known. After a call to EnableAlerts() this gets set to the value returned from the agent, which may be Unavailable, Enabled or Disabled.

Type

Property name	Type
Description	"The Type property is the type of device"
Type	String
Access	Read, Write
Key	No
Values	"Microsoft Server Appliance"

Description

Property name	Description
Description	"Description of device"
Type	String or NULL
Access	Read, Write
Key	No
Values	256 Unicode characters

LastDiscoveryTime

Property name	LastDiscoveryTime
Description	"Time when this controller device last received auto-discovery information from the controlled device. The value is NULL if the controlled device has never been auto-discovered."
Type	Datetime or NULL
Access	Read
Key	No
Values	

Every time that an auto-discovery packet is received that matches this Devices instance, that date is updated to the current date. This can be used to identify manually entered records that have never been seen on the network.

The following properties are inherited from the parent classes but are presently unused: Caption, CreationClassName, InitialLoadInfo, InstallDate, LastLoadInfo, NameFormat PowerManagementSupported, PowerManagementCapabilities, PowerState, PrimaryOwnerContact, PrimaryOwnerName, ResetCapability, Roles, Status and Time.

Methods

EnableAlerts

Description	"The EnableAlerts method determines whether a device sends Server Appliance Kit alerts to the controller device..	
Arguments:		
EnableFlag	[in] Boolean	"The EnableFlag input parameter is a Boolean value that enables or disables the alerts for True or False flag values correspondingly."
Return value:	Void	

This call is synchronous. If alerts are enabled on a device, the device will return the current alerts. If alerts are disabled, the controller will delete the details of alerts for this device.

RecoverManagedDevice

Description	"The RecoverManagedDevice method returns devices to normal state. This method can be invoked to bring back to the normal operating state if the controlled device does not respond to requests from the controller device.."	
Arguments:		
Return Value:	Void	

This method can be called if the user believes that the configuration on a device is corrupted. The controller sends out the following information to the device:

- A request for the device information.
- A control request to manage this device
- The current heartbeat configuration information (the heartbeat interval)
- The current alert status for this device (enabled or not enabled)

This call returns after all of these have been processed, and thus could take some time.

Manage

Description	"The Manage method places devices into either a controlled or an uncontrolled state."	
Arguments:		
ControlFlag	[in] 32 bit integer	The ControlFlag input parameter is the value that specifies how to manage the device."
Return value:	Void	

This changes the state of control for a device, and returns when that is complete. The value of ControlFlag specifies the operation to perform against the device:

Value of ControlFlag	Meaning	Description
0	"Release control"	The device becomes uncontrolled. The controller's root certificate is left on the device, so the device can only be controlled by another controller with the same root certificate.
1	"Take control"	The device becomes controlled from this controller.
2	"Release control and remove certificate"	The device becomes uncontrolled. The controller's root certificate is deleted from the device, so the device can be controlled by any controller.

An Error is returned if an attempt to control a device fails because of a protocol version error.

Execute

Description:	"The Execute method executes a command on devices, and then returns the job identifier relating to the parent job."	
Arguments:		
JobInvocationName	[in] String	"JobInvocationName input parameter is the name of the JobInvocation instance that will be created."
CommandType	[in] 32 bit integer	See Sets.Execute
Command	[in] String	"Command input parameter is the path of the command to be executed."
Parameters	[in] String	"Parameters input parameter is the parameters to be passed to the executing command."
Description	[in] String	"Description input parameter is the description for the executing command which is to be logged

		in the JobInvocation object”
JobID	64 bit integer	JobID of the new job, or 0 if an error occurred
Return value:	Void	

The SetPowerState method is inherited from the parent classes but is currently unused in this implementaion.

DeviceTypes

In one implementation, devices report the same device type, “Microsoft Server Appliance”.

Class name	DeviceTypes
Derived from	CIM LogicalElement
Description	“The DeviceTypes class represents the various possible device types on the network.”

Instances of DeviceTypes can be created manually or automatically. When created manually, the Name needs to be unique across existing DeviceTypes instances. Manual creation can be used on networks where auto-discovery does not work.

Instances of DeviceTypes may be created automatically based on incoming auto-discovery packets, or on creation of instances of Devices. Another way that instances of DevicesTypes are created is based on the creation of Devices instances. Each Devices instance includes a DeviceType string. If a DeviceTypes instance does not exist corresponding to this string, a new instance of DeviceTypes is created with this string as its Name, and a blank Description.

Deletion

Instances of DeviceTypes can be deleted, but only if there are no instances of Devices with the same type string as the one being deleted. If deletion fails because of this, the WMI DeleteInstance method should return with error WBEM_E_FAILED.

Associations

An instance of DeviceTypes is associated with each instance of Devices that is of the same type.

Properties

Name

Property name	Name
Description	"The Name property is the unique identifier for the device that is available on the network."
Type	String
Access	Read
Key	Yes
Values	50 Unicode characters

The name of a device type cannot be changed.

Description

Property name	Description
Description	"Description of the devicetype."
Type	String
Access	Read, Write
Key	No
Values	256 Unicode characters

The Caption, InstallDate and Status properties are inherited from the parent classes but are currently unused in a current implementation. There are no methods for this object.

HWADDRTYPES

Every device typically contains many individually identifiable hardware parts. Each hardware part can contain an address or unique identifier.

Class name	HWAddrTypes
Derived from	CIM LogicalElement
Description	"The HWAddrTypes class represents the possible types of device parts on the network."

In one version, the only address type used internally is "MAC". This is used to store the MAC addresses of devices. Instances of HWAddrTypes can be manually or automatically created. An instance will be automatically created if an instance of DeviceHWAddrs is created where the Type property value does not match the Name property of an existing HWAddrTypes instance. In this case, the new instance of HWAddrTypes will be created with a Name property having the same value as in the DeviceHWAddrs Type property, and the Description property will be blank.

Deletion

Instances of HWAddrTypes can be deleted. The deletion will fail if any Devices instances contain the same Type property as the Name property on the instance being deleted.

Associations

An instance of HWAddrTypes is associated with all instances of DeviceHWAddrs that contain hardware address information for this type of hardware.

Properties

Name

Property name	Name
Description	"The Name property is the unique identifier for the device parts that is available on the network."
Type	String
Access	Read
Key	Yes
Values	50 Unicode characters

Description

Property name	Description
Description	"The Description property is the description of the device part."
Type	String
Access	Read, Write
Key	No
Values	256 Unicode characters

The Caption, InstallDate and Status properties are inherited from the parent classes but are currently unused in a current implementation. There are no methods for this object.

DEVICEHWADDRS

The device hardware address uniquely identifies a piece of hardware such as a NIC.

Class name	DeviceHWAddrs
Derived from	CIM LogicalElement
Description	"The DeviceHWAddrs class represents the device parts and their hardware addresses."

In one current implementation, the only type of hardware address that is used internally is "MAC". This is used to store the MAC addresses of devices. Instances of DeviceHWAddrs can be created automatically or manually. Automatic creation occurs based on received auto-discovery packets. If the packet contains a hardware address (containing a type and address), then a new instance of DeviceHWAddrs is created for the hardware address (which may also involve creating an instance of HWAddrTypes

Deletion

Instances of DeviceHWAddrs can be deleted. Instances are also deleted automatically when the corresponding Devices instance is deleted.

Associations

- To single instance of Devices

  An instance of DeviceHWAddrs is associated to the instance of Devices which contains has the hardware address.

- To single instance of HWAddrTypes

  An instance of DeviceHWAddrs is associated to the instance of HWAddrTypes which defines the type of hardware address stored in this DeviceHWAddrs instance.

- To instances of DeviceHWIPAddrs

  An instance of DeviceHWAddrs is associated with zero or more instances of DeviceHWIPAddrs for each IP address that is associated with this instance of DeviceHWAddrs. This association is only used for DeviceHWAddrs that contain the address of NIC hardware.

Properties

HWAddr

Property name	HWAddr
Description	"The HWAddr property is the unique identifier for the device parts."
Type	String
Access	Read
Key	Yes
Values	50 Unicode characters

DeviceName

Property name	DeviceName
Description	"DeviceName property is the name of the device that holds the device part."
Type	String
Access	Read
Key	No
Values	A device name. This is the name of the instance of Devices corresponding to this hardware address.

Type

Property name	Type
Description	"Type property is the type of hardware address."
Type	String
Access	Read
Key	No
Values	50 Unicode characters.

Unused Properties

The Caption, InstallDate, Name and Status properties are inherited from the parent classes but are currently unused in a current implementation. There are no methods for this object.

DEVICEHWIPADDRS

This object is not presently used

JOBINVOCATIONS

There are two categories of Jobs Invocations. A first category is templates. These are Jobs Invocations with RootJobID equal to zero, meaning that these are just templates and not associated with a particular run. The second category of Jobs Invocations are historical, meaning that they are a historical record of a job invocation.

Class name	JobInvocations
Derived from	CIM Job
Description	"The JobInvocations class represents either a job template or a previously run job, which is called a job history. Job templates have a value of 0 in RootJobID, while previously run jobs have a non-zero RootJobID."

Instances of JobInvocations can be created manually or automatically. Job templates are created manually, while Job historical records are created automatically. Instances of JobInvocations can be manually created by the user (or other process). Only job templates may be created (by definition, instances of JobInvocations where RootJobID is zero), and it is an error to try to create a JobInvocations instance with a RootJobID of other than zero. Manually created instances of JobInvocations may later have any of its writable properties changed.

Automatic Creation (Job Histories)

When a job is executed (using Devices.Execute or Sets.Execute), an instance of JobInvocations is created. This will be given a new unique value of RootJobID (which will not be zero). The properties Name, Command, Parameters and Description of the new instance will be populated with values of the JobInvocationName, Command, Parameters and Description arguments to the Devices.Execute or Sets.Execute method that created the job. The properties TargetName and TargetType will be filled in with the name of the set or device on which the job is being executed, and the type of set or device. Properties of an automatically created JobInvocations instance may not be changed. In WMI, attempting to change a property of a JobInvocations instance will cause the PutInstance (Put_ from script) method to return WBEM_E_FAILED (WbemErrFailed from scriptOnly job templates (instances where RootJobID is zero) and job histories (where RootJobID is non-zero) can both be deleted. If a job history is deleted, all the associated Jobs and JobLogs instances are also deleted. This is implemented in WMI.

Associations

- To a single instance of Devices

  An instance of JobInvocations is associated with an instance of Devices if the job template is defined to be run on that device, or if the job history was run on that device. This association exists only if the value of TargetType is Devices, and if so, the association is to the Devices instance with the same name as given in the TargetName property.

- To a single instance of Sets

  An instance of JobInvocations is associated with an instance of Sets if the job template is defined to be run on that set, or if the job history was run on that set. This association exists only if the value of TargetType is Sets, and if so, the association is to the Sets instance with the same name as given in the TargetName property.

- To a single instance of Jobs

  An instance of JobInvocations is associated with an instance of Jobs, which gives the results of running the job represented by JobInvocations. This association only exists if the JobInvocations instance is a historical record (that is, RootJobID is non-zero).

Properties

RootJobID

Property name	RootJobID
Description	"The RootJobID property works with the Name property to uniquely identify jobs. The RootJobID value is zero for job templates, and nonzero for job histories."
Type	64 bit integer
Access	Read
Key	Yes
Values	0 = this is a job template non-zero = unique identifier for a previously executed job (assigned by Devices.Execute or Sets.Execute)

Root job unique identifier. This can be any 64 bit value. If it is zero, it indicates that this JobInvocations instance is a template rather than a historical record for a previous executed job. If this value is non-zero, then this instance is a historical record.

Name

Property name	Name
Description	"The Name property is the identifier for the JobInvocations object."
Type	String
Access	Read
Key	Yes
Values	50 Unicode characters

This stores the name of the job as passed as the JobInvocationName argument to Devices.Execute or Sets.Execute.

TargetName

Property name	TargetName
Description	"Name of the target, such as sets or devices, on which the job was invoked."
Type	String
Access	Read, Write
Key	No
Values	256 Unicode characters

Target name the job was run on: Devices or Sets.

TargetType

Property name	TargetType
Description	"Type of the target, such as sets or devices."
Type	32 bit integer
Access	Read, Write
Key	No
Values	1 = Sets, 2 = Devices

Command

Property name	Command
Description	"The job command that is to be executed on the target object."
Type	String
Access	Read, Write
Key	No
Values	1024 Unicode characters

The command that is invoked on the target.

Parameters

Property name	Parameters
Description	"Parameters passed to the job command that is to be executed."
Type	String
Access	Read, Write
Key	No
Values	4096 Unicode characters

The parameters to the command that is invoked.

Description

Property name	Description
Description	"Description of the job that was invoked."
Type	String
Access	Read, Write
Key	No
Values	256 Unicode characters

Unused Properties

The following properties are inherited from the parent classes but are currently unused: Caption, ElapsedTime, InstallDate, Notify, Owner, Priority, StartTime, Status, TimeSubmitted, UntilTime

Methods

Rename

Description	"The rename method renames it to the name specified.X"	
Arguments:		
InvocationName	[in] String	"The InvocationName input parameter is the name to which the JobInvocations object is to be renamed."
Return value:	Void	

JOBS

The Jobs object captures the topology of a Job Invocation. For example, if the user had a set object and called Execute on that set, then there would be one Jobs Object created as the Root Job and also a Jobs object created for each device of the set. The Root Jobs object is used as the entry into the topological structure of the Jobs. In one implementation only use the parent child relation is used, however, the model is capable of instantiated an N deep tree.

Class name	Jobs
Derived from	CIM LogicalElement
Description	"The Jobs class displays the results of an executed job. An instance of this class is created for each job that is run and for each device on which a job is run.."

Jobs are automatically created by the system, when a job is started (for example, by the methods Sets.Execute or Devices.Execute).

Instances of Jobs cannot be updated. If an instance of Jobs that represents a parent job is deleted, the following are also deleted:

- The associated JobInvocations instance
- The child instances of Jobs associated with the Jobs instance
- The JobLogs instances associated with each of the child Jobs instances

This deletes the record of the job invocation, the status of the job on each device, and the results of the jobs on each device.

Jobs currently in progress can be deleted. If this occurs, any further output from the job received on the controller will not be stored, and no error will be reported.

Associations

- To a single instance of JobInvocations

  An instance of Jobs is associated to the instance of JobInvocations that represents the job.

- To instances of JobLogs

  An instance of Jobs is associated to instances of JobLogs that contain the output from running this job on a particular device. This association only exists if the instance of Jobs represents a single device, rather than a parent instance.

- To instances of Jobs

An instance of Jobs is associated to instances of Jobs representing the child processes of the parent Jobs instance. In the current version, only one level of parent-child relationship is supported, where the parent represents a job being run on multiple devices and the children represent the results from individual devices.

- To a single instance of Devices

  An instance of Jobs is associated to a instance of Devices when the Jobs instance holds the results of running a job on a single device. The association gives the device that the job is run on.

Properties

JobID

Property name	JobID
Description	"The JobID property is the unique identifier for the job that has been executed. If the same job executes again, it yields a different identifier."
Type	64 bit integer
Access	Read
Key	Yes
Values	Job unique identifier

ParentJobID

Property name	ParentJobID
Description	"The ParentJobID property is the identifier for the job which originates the job on the targets. For the root job, the identifier will be 0 and for other jobs, the identifier uses the identifier of the root job."
Type	64 bit integer
Access	Read
Key	No
Values	Parent job unique identifier

DeviceName

Property name	DeviceName
Description	"Name of the device on which the job was run."
Type	String
Access	Read
Key	No
Values	Name of device job was run on.

StartTime

Property name	StartTime
Description	"Time stamp when task was started."
Type	Datetime
Access	Read
Key	No
Values	The time the job was started

EndTime

Property name	EndTime
Description	"Time at which this task was completed."
Type	Datetime
Access	Read
Key	No
Values	The time the job was completed

JobStatus

Property name	JobStatus
Description	"Indicates the status of the task."
Type	32 bit integer
Access	Read
Key	No
Values	0 = "Job completed" 1 = "Job completed with errors" 2 = "No target to send" 3 = "Job started" 4 = "Send completed" 5 = "Job failed" 6 = "Job stopped by user" 7 = "Send failed" 8 = "Send not started" 9 = "Uncontrolled device" 10 = "Inactive device" 11 = "Job running with errors"

When the Sets.Execute or Devices.Execute method returns, the value of the status will be one of 2, 3 or 11. If the job cannot be started on any devices, the status is set to 2. If the job cannot be started on some of the devices, the status is set to 11. Otherwise the job was started on all devices, and the status is set to 3.

After the Execute method has returned, the caller can poll the value of the status for the parent Jobs instance. If the job is still running on at least one device, the status will be 3 or 11. If the job has finished on all devices, the status will be 0 or 1. If the job never started on any devices, the status will be 2.

For child jobs, valid values are 0 and 3 through 10. It is initially set to 3 when the child Jobs object is created.

If an error occurs running the job, the JobStatus will be updated to one of 5 or 7 to 10. If the job is successfully transmitted to the device, the JobStatus will be set to 4. If the job is stopped by the user calling Jobs.Stop, the JobStatus will be set to 6. If the job completes successfully, JobStatus will be set to 0.

The following properties are inherited from the parent classes but are currently unused: Caption, Description, InstallData, Name and Status.

Methods

Stop

Description:	"The Stop method stops a job that is in progress."	
Arguments:	None	
Return value:	Void	

This stops a job executing on a device. If it is executed on an instance that represents a parent job, all the child chills that are still running are stopped. A job that was stopped by the user will have a Status property value of 'Job Stopped by User'.

GetOutput

Description:	"GetOutput method retrieves the output from the job log and yields the collective result."	
Arguments:		
OutputType	[in] 32 bit integer	"Type of the output to be retrieved from the JobLogs."
Output	[out] String	[See DCR 4988]
Return value:	Void	

The values for OutputType are:

- 0 = Get exit status
- 1 = Get standard output
- 2 = Get standard error
- 3 = Get all output (in sequence order)

This method is only valid for child Job instances. If OutputType is 0 the exit status is returned in the Output string. For example an exit status of 32 would be returned as the string "32" in Output.

JOBLOGS

JobLogs capture the output of the execution of a script or executable. The job log is associated with a Jobs. There can be N JobLogs for any Jobs.

Class name	JobLogs
Derived from	CIM LogicalElement
Description	"The JobLogs class represents the output log for the jobs that have already been executed."

JobLogs are always auto-created by the system, when a job is started (for example, by the methods Sets.Execute or Devices.Execute).

Instances of the JobLogs class cannot be updated, and instances of JobLogs can not be deleted in WMI. Associations are to a single instance of Jobs; an instance of JobLogs is associated with an instance of Jobs giving the device and job which generated this output.

Properties

JobID

Property name	JobID
Description	"The JobID property is the unique identifier for the job that has been executed."
Type	64 bit integer
Access	Read
Key	Yes
Values	Jobs unique identifier

Sequence

Property name	Sequence
Description	"Sequence of the output from the job that was executed on the device under consideration."
Type	32 bit integer
Access	Read
Key	Yes
Values	Record sequence for a jobs output

Starts at 1 going up. Same sequence number used for stdout, stderr, exit status.

LogTime

Property name	LogTime
Description	"Time at which the controller device received output."
Type	Datetime
Access	Read
Key	No
Values	

OutputType

Property name	OutputType
Description	"The OutputType property specifies the type of output in this instance of JobLogs. It is one of: 0 meaning ExitCode, 1 meaning StdOut,or 2 meaning StdErr."
Type	32 bit integer
Access	Read
Key	No
Values	0 = "ExitCode" 1 = "StdOut" 2 = "StdErr"

OutputData

Property name	OutputData
Description	"Output from the job on the device. The sequence property can be used to recreate the output from this job on this device in correct order."
Type	String
Access	Read
Key	No
Values	

These properties are inherited from the parent classes but are currently unused: Caption, Description, InstallDate, Name and Status. There are no methods for this object.

ALERTS

The device hardware address uniquely identifies a piece of hardware such as a NIC.

Class name	Alerts
Derived from	CIM LogicalElement
Description	"The Alerts class represents a Server Appliance Kit alert on a device."

The alerts object maintains copies of all the alerts from managed devices which are reporting alerts to the controller. Whether a device reports alerts or not is set using the Devices.EnableAlerts() method. Each instance of the Alerts object represents a single alert from a single device. Alert instances are always created automatically, based on information provided from devices. They cannot be manually created (in WMI).

Instances of this class cannot be updated, and deleting an instance of the alerts class causes the alert to be cleared on the device itself.

Associations are to a single instance of Devices; to the instance of Devices that represents the device on which the alert was created.

Properties

DeviceName

Property name	DeviceName
Description	"The device from where the alert is fired."
Type	String
Access	Read
Key	Yes
Values	TBD

This contains the name of the device on which the alert was created.

Cookie

Property name	Cookie
Description	"Cookie for the alert."
Type	32 bit integer
Access	Read
Key	Yes
Values	TBD

AlertType

Property name	AlertType
Description	"Type of the alert."
Type	String
Access	Read
Key	No
Values	TBD

AlertID

Property name	AlertID
Description	"Identifier of the alert."
Type	32 bit integer
Access	Read
Key	No
Values	TBD

AlertLog

Property name	AlertLog
Description	"Name of the alert log."
Type	String
Access	Read
Key	No
Values	TBD

AlertSource

Property name	AlertSource
Description	"Source of the alert."
Type	String
Access	Read
Key	Yes
Values	TBD

AlertReplaceString

Property name	AlertReplaceString
Description	"Replacement strings of the alert."
Type	String
Access	Read
Key	No
Values	Any length strings

This is stored in the format as received from the device. At present, this is as a list of values separated by the line feed character (character code 10 decimal). No escaping is performed in case the values contain commas.

ReceivedTime

Property name	ReceivedTime
Description	"Time at which the alert is fired."
Type	Datetime
Access	Read
Key	No
Values	TBD

This is the time at which the controller received the alert notification from the device.

The following properties are inherited from the parent classes but are currently unused: Caption, Description, InstallDate, Name and Status. There are no methods for this object.

CONTROLLER

The controller class is used to configure and control the controller. The controller has several global configurable parameters such as heartbeat interval. Additionally, the controller service and sub-services can be started and stopped.

Class name	Controller
Derived from	CIM Configuration
Description	"The Controller class represents the configuration data store that relates to the controller device."

This is a singleton class with respect to creation and deletion, and this instance cannot be updated. There are no associations. Properties

HeartbeatInterval

Property name	HeartBeatInterval
Description	"The HeartbeatInterval property is the heartbeat communication interval between the controlled device and the controller device.."
Type	String
Access	Read
Key	No
Values	Number of seconds between heartbeats.

If not specified, the value 120 seconds (2 minutes) is used. The minimum it can be set to is 60 seconds. The following properties are inherited from the parent classes but are currently unused: Caption, Description and Name.

Methods

RefreshDeviceList

Description	"The RefreshDeviceList method refreshes the list of the devices that have already been detected or manually added. It first clears the list of uncontrolled devices, and then broadcasts multicast SSDP discovery request. Upon receiving the unicast response from the appliances on the network, the method populates the list of appliances."
Arguments:	none
Return value:	Void

Refresh device list is a two step process. First, uncontrolled devices are removed from the data store. Then a solicited discovery is initiated for devices with the type "Microsoft Server Appliance. The method then returns. At this point, the results of initiating the discovery may not have been received by the controller, so the Devices table may be empty or partially complete.

SetHeartbeatInterval

Description	"The SetHeartbeatInterval method sets the interval at which heartbeat communication is to be made."	
Arguments:		
Interval	[in] 32 bit integer	"The Interval property is the interval in seconds at which the heartbeat communication is to happen."
Return value:	Void	

This updates the heartbeat interval in the Controller.HeartbeatInterval property. It then sends this updated interval out to all the managed devices. The method then returns. It does not wait for status of sending the updated heartbeat to the devices. This value is given in seconds. The minimum it can be set to is 60. If an attempt is made to set it less than 60, the stored value will be set to 60 in one implementation.

Note that if this if this is increased, devices could appear to be not-alive after 1.5*old-heartbeat-interval (the Devices.Alive property will be set to false). In WMI, special classes called association classes are used to link instances of objects. This section defines the association classes used to link the WMI classes that implement the object model.

Association classes are derived from either CIM_Component or CIM_Dependency.

CIM_Component contains the properties GroupComponent and PartComponent to specify the parent instance and child instance respectively. CIM_Dependency contains the properties Antecedent and Dependent to specify the dependency relationship. As described below, the parent class is listed, then the values of the properties (either GroupComponent and PartComponent, or Antecedent and Dependent) are described. No classes add additional properties or override the descriptions or other attributes.

DEVICEHWADDRTODEVICEHWIPADDR

Class name	DeviceHWAddrToDeviceHWIPAddr
Derived from	CIM Component
Description	<inherited>

GroupComponent is a reference to an instance of DeviceHWAddrs, PartComponent is a reference to an instance of DeviceHWIPAddrs.

DEVICEHWADDRTOHWADDRTYPE

Class name	DeviceHWAddrToHWAddrType
Derived from	CIM Dependency
Description	<inherited>

Antecedent is a reference to an instance of DeviceHWAddrs, Dependent is a reference to an instance of HWAddrTypes.

DEVICETOALERT

Class name	DeviceToAlert
Derived from	CIM Dependency
Description	<inherited>

Antecedent is a reference to an instance of Devices, Dependent is a reference to an instance of Alerts.

DEVICETODEVICEHWADDR

Class name	DeviceToDeviceHWAddr
Derived from	CIM Component
Description	<inherited>

GroupComponent is a reference to an instance of Devices, PartComponent is a reference to an instance of DeviceHWAddrs.

DEVICETODEVICETYPE

Class name	DeviceToDeviceType
Derived from	CIM Dependency
Description	<inherited>

Antecedent is a reference to an instance of Devices, Dependent is a reference to an instance of DeviceTypes.

DEVICETOJOBINVOCATION

Class name	DeviceToJobInvocation
Derived from	CIM Dependency
Description	<inherited>

Antecedent is a reference to an instance of Devices, Dependent is a reference to an instance of JobInvocations.

SETTOJOBINVOCATION

Class name	SetToJobInvocation
Derived from	CIM Dependency
Description	<inherited>

Antecedent is a reference to an instance of Sets, Dependent is a reference to an instance of JobInvocations.

DEVICETOJOB

Class name	DeviceToJob
Derived from	CIM Dependency
Description	<inherited>

Antecedent is a reference to an instance of Devices, Dependent is a reference to an instance of Jobs.

JOBTOJOB

Class name	JobToJob
Derived from	CIM Component
Description	<inherited>

GroupComponent is a reference to an instance of Jobs (representing the parent), PartComponent is a reference to an instance of Jobs (for the children).

JOBINVOCATIONTOJOB

Class name	JobInvocationToJob
Derived from	CIM Dependency
Description	<inherited>

Antecedent is a reference to an instance of JobInvocations, Dependent is a reference to an instance of Jobs.

JOBTOJOBLOG

Class name	JobToJobLog
Derived from	CIM Dependency
Description	<inherited>

Antecedent is a reference to an instance of Jobs, Dependent is a reference to an instance of JobLogs.

SETTODEVICE

Class name	SetToDevice
Derived from	CIM Component
Description	<inherited>

GroupComponent is a reference to an instance of Sets, PartComponent is a reference to an instance of Devices.